한국지능시스템학회 논문지 2009, Vol. 19, No. 2, pp. 224-229

침입자 포위를 위한 군집 로봇의 분산 이동 알고리즘

Distributed Moving Algorithm of Swarm Robots to Enclose an Invader

이희재 · 심귀보*

Hea-Jae Lee and Kwee-Bo Sim*

중앙대학교 전자전기공학부

요 약

군집 로봇(swarm robots)이 같은 작업 환경에 존재할 때, 우리는 어떤 임무를 수행하기 위한 로봇들을 먼저 결정해야 한다. 이런 로봇들의 협조 행동을 제어하기 위한 연구들이 많이 있었다. 이런 군집 로봇 시스템을 사용함으로써 얻는 이점은 협조 행동을 통해서 임무 수행의 적응성과 융통성이 증가하는 특성이라 할 수 있다. 침입자가 발견 되었을 때, 군집 로봇은 효율적인 포위를 위해서 침입자의 이동 경로를 예상하면서 다양한 경로를 통해서 침입자에게 접근, 포위해야 한다. 본 논문에서는 2차원 맵에서의 군집 로봇의 효율적인 포위 방법과 분산 이동 알고리즘을 제안한다.

Abstract

When swarm robots exist in the same workspace, first we have to decide robots in order to accomplish some tasks. There have been a lot of works that research how to control robots in cooperation. The interest in using swarm robot systems is due to their unique characteristics such as increasing the adaptability and the flexibility of mission execution. When an invader is discovered, swarm robots have to enclose a invader through a variety of path, expecting invader's move, in order to effective enclose. In this paper, we propose an effective swarm robots enclosing and distributed moving algorithm in a two dimensional map.

Key Words : Swarm robots, Distribute moving algorithm, Enclose an invader

1. 서 론

군집 로봇의 협조 행동을 제어하는 것은 크게 2가지 제어로 나눌 수 있다. 첫 번째가 모델에 기반한 제어로서 우리가 알고 있는 정확한 kinematics와 dynamics를 이용한 모델링에 의한 제어를 말한다. 두 번째는 행동에 기반한 제어로서 우리가 정확한 모델을 알고 있지 못하며, 어떤 행동을 특정 알고리즘으로서 나타내는 것이다[4]. 본 논문에서는 후자에 초점을 두고 있다.

군집 로봇의 협조 행동 중에서 침입자를 포위하는 연구는 최근 10년간 활발히 연구되고 있다[1][4][5]. 전자의 모델링에 의한 군집 로봇의 제어 연구로는 Antonelli의 연구[2][3]가 있으며, 후자의 연구로는 Kamano가 퍼지 룰을 이용하여 로봇이 협조적으로 타겟을 포위하는 방법을 제안하였고[7], Pirjanian은 실험에 근거한 방법을 제안하였다[6]. 침입자를 포위하는 행동을 특정 알고리즘으로서 나타내기 위해서는 먼저 이동 로봇들을 제어하는 시스템의 local input과 global output을 생각해야 한다[5]. 여기서 local input은 각각의 로봇들의 위치 좌표와 침입자의 위치 좌표가 되며, global output은 로봇들이 침입자를 포위하였을 때의 최종 위치와 그 위치까지의 이동 경로가 된다. 먼저 로봇의 포메이션은 침입자의 탈출을 효과적으로 막을 수 있어야 한다. 이러한 포메이션 중 하나로 로봇들이 침입자가 이동할 확률이 높은 방향을 예측해 그 방향을 차단하는 방법을 생각할 수 있다. 침입자가 이동할 확률이 높은 방향을 예측하기 위해서는 주어진 맵을 분석하여야 한다. 우리는 침입자가 탈출이 가능한 장소, 예를 들어 문과 창문 같은 장소에 탈출 가능성에 따라서 직관적으로 확률 테이블을 구성하고, 침입자의 위치와 장애물의 위치 등의 요소를 고려하여 로봇은 침입자의 예상 이동 경로를 차단하는 곳에 위치하며 침입자를 포위하는 것이 효율적인 로봇의 포메이션이라고 정의하고 군집 로봇의 포메이션 알고리즘을 구성하였다. 또한 로봇들은 침입자를 포위하는 이동시에 서로 협조적으로 침입자의 예상 이동 경로를 차단하며 침입자에게 접근해야 한다. 이러한 것을 위하여 본 논문에서는 군집 로봇 이동 제어 시스템은 침입자의 예상 이동 경로를 확률로서 계산하여 주어진 포위 위치까지 로봇들의 분산 이동 경로를 계획하는 알고리즘을 제안한다.

2장에서는 군집 로봇의 제어 시스템의 구조에 대해 기술하고, 3장에서는 포메이션 알고리즘, 4장에서는 경로 계획 알고리즘, 마지막으로 5장에서는 시뮬레이션 및 그 결과에 대해서 기술한다.

접수일자 : 2008년 9월 19일
완료일자 : 2009년 4월 6일
* 교신 저자
본 연구는 지식경제부의 2008년도 성장동력기술개발사업인 「집단 로봇 기술을 이용한 사회안전로봇 개발(세부과제: 로봇통제 및 환경기술개발)」에 의해 수행되었습니다. 연구비 지원에 감사드립니다.

2. 군집 로봇 제어 시스템의 구조

우리는 알려진 2차원 맵에서 군집 로봇의 협조 행동을 통해 침입자를 포위하는 알고리즘을 위해서 그림 1과 같은 군집 로봇 제어 시스템 구조를 설계하였다. 먼저 우리는 군집 로봇은 모두 같은 능력을 가지고 있다고 가정하였다. 침입자 발견 시 제어 시스템은 먼저 모든 로봇들의 현재 위치와 침입자의 위치를 입력받아야 한다. 이 전송된 맵 정보를 가지고 제어 시스템은 포메이션 알고리즘으로서 로봇의 최종 포메이션 좌표를 결정한다. 이 정보를 다시 경로 계획 알고리즘에서 입력받아 포메이션 알고리즘에서 결정된 최종 좌표와 이동 경로를 각각의 로봇에 계획해준다. 이동 경로는 주어진 2차원 맵을 장애물의 위치를 고려하여 각 영역으로 구분하여 주며, 또한 이 영역에서의 각각의 포인트를 지정한다. 이 포인트라는 것은 여러 대의 로봇이 동시간대에 해당 영역을 지나가게 될 때에 혼잡을 방지하기 위하여 만들어진 경유 좌표이다. 이것은 차후 4장 경로 계획 알고리즘에서 자세히 기술될 것이다.

제어 시스템은 각각의 로봇에게 이동할 방향 벡터를 전송하며, 방향 벡터는 식(1)과 같이 정의할 수 있다. 여기서 $[x_{point}, y_{point}]^T$는 위에서 언급한 경유 좌표 또는 마지막 포메이션 좌표이다. 그리고 $[x_{robot}, y_{robot}]^T$는 현재 로봇의 위치 좌표이다.

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} x_{point} \\ y_{point} \end{bmatrix} - \begin{bmatrix} x_{robot} \\ y_{robot} \end{bmatrix} \tag{1}$$

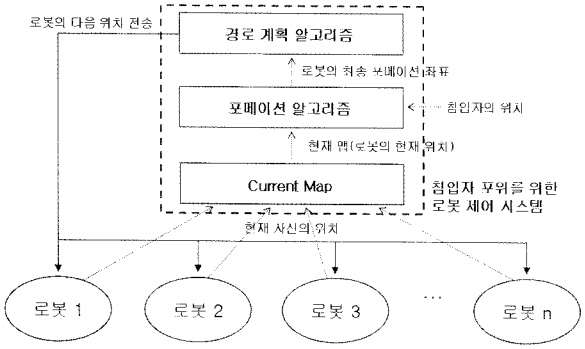

그림 1. 군집 로봇 제어 시스템의 구조
Fig 1. The architecture of control system for swarm robots

3. 포메이션 알고리즘

3장에서는 로봇의 최종 포메이션을 위한 포메이션 알고리즘을 제안한다. 로봇과 침입자의 위치 좌표, 그리고 맵 정보를 가지고 포메이션 알고리즘은 로봇의 최종 포메이션을 결정해준다. 먼저 그림 2와 같은 2차원 맵에서 침입자가 발견 되었을 때, 우리는 침입자가 이동할 방향을 확률로서 나타내기 위해 문이나 창문 등 직관적으로 탈출이 가능하다고 판단되어지는 장소에 다음과 같은 초기 적합도를 주었다.

$$Fitness(Door) = 5$$
$$Fitness(Window) = 1$$

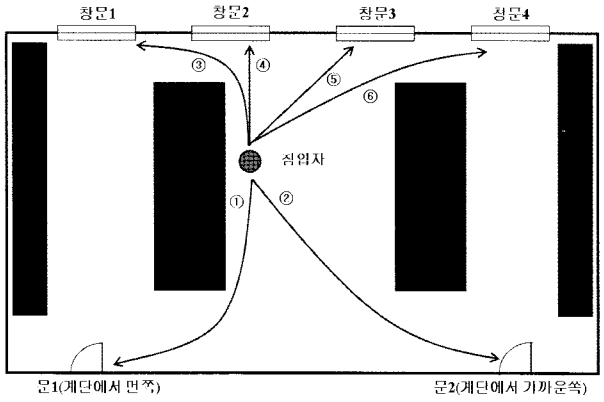

그림 2. 직관적으로 판단한 침입자의 예상 탈출 경로
Fig 2. The intuitive expected escapes of an invader

여기서 문 밖에 계단이 있으면 적합도를 +1해주었고, 1층인 경우에는 창문에 +7의 적합도를 부여하였다. 이 적합도는 직관적으로 결정해준 적합도이며, 이 적합도의 최적의 값을 찾는 일은 쉽지 않다. 향후 연구에서는 학습을 통하여 적합도를 조절할 수 있는 알고리즘을 수정할 계획이다. 이렇게 정해준 초기 적합도에 따라 해당 장소로의 침입자 탈출 확률이 다음과 같이 결정된다.

$$P(s_i) = Fitness(s_i) / \sum_{j=1}^{n} Fitness(s_j) \tag{2}$$

여기서 n은 문이나 창문 등 탈출이 가능한 장소의 개수이며, 식(2)로부터 그림 2의 각 장소로 이동할 확률을 구해보면 다음과 같다.

$$P(door1) = 5/(5+6+1+1+1+1) = 0.333$$
$$P(door2) = 6/(5+6+1+1+1+1) = 0.40$$
$$P(window1) = P(window2)$$
$$= P(window3) = P(window4) = 0.067$$

우리는 침입자에서 탈출이 가능한 장소로의 방향 $[x_{m(n)}, y_{m(n)}]^T$을 각각의 탈출이 가능한 장소의 위치 좌표를 가지고 1의 과정을 통해 얻을 수 있다.

Step ① :

for $i = 1$ *to* n

$$\begin{bmatrix} x'_{m(n)} \\ y'_{m(n)} \end{bmatrix} = \begin{bmatrix} x_{site(n)} \\ y_{site(n)} \end{bmatrix} - \begin{bmatrix} x_{invader} \\ y_{invader} \end{bmatrix}$$

$$\begin{bmatrix} x_{m(n)} \\ y_{m(n)} \end{bmatrix} = \frac{1}{c} \begin{bmatrix} x'_{m(n)} \\ y'_{m(n)} \end{bmatrix},$$

where c *is* $c \in N, c > 0$

end

여기서 m대의 로봇이 침입자를 포위하고 $l(a < l < b)$은 침입자와의 거리라고 가정했을 때, 로봇의 최종 포메이션에서의 l은 사용자가 지정한 파라미터 a보다 크고 b보다 작아야한다. 이와 같은 조건을 만족하기 위해서 다음과 같이 ②의 과정을 수행한다.

Step ② :

for $i = 1$ *to* n

$k_1 = 2,\ k_2 = 0.5,\ m_1 = 0.5,\ m_2 = 0.25$

$while(1)$

if $a < \|[x_{m(i)}, y_{m(i)}]^T\| < b,$

$$\begin{bmatrix} x_{location(i)} \\ y_{location(i)} \end{bmatrix} = \begin{bmatrix} x_{invader} \\ y_{invader} \end{bmatrix} - \begin{bmatrix} x_{m(i)} \\ y_{m(i)} \end{bmatrix}$$

break

elseif $\| [x_{m(i)}, y_{m(i)}]^T \| < a$,

$x_{m(i)} = [k_1 x_{m(i)}]$, $y_{m(i)} = [k_1 y_{m(i)}]$

elseif $\| [x_{m(i)}, y_{m(i)}]^T \| > b$,

$x_{m(i)} = [k_2 x_{m(i)}]$, $y_{m(i)} = [k_2 y_{m(i)}]$,

$k_1 = k_1 - m_1$, $k_2 = k_2 + m_2$

$m_1 = 0.5 m_1$, $m_2 = 0.5 m_2$

end

여기서 $[\bullet]$은 $\bullet$ 보다 작은 정수이다. 이렇게 위 알고리즘으로부터 얻은 $[x_{location(n)}, y_{location(n)}]^T$이 로봇의 최종 위치 좌표이며, 이 좌표의 수는 n개 이다. 하지만 로봇의 수는 m대이므로 우리는 위에서 얻은 최종 위치 좌표의 수n을 로봇의 수 m과 맞춰주어야 한다. 이를 위하여 먼저 우리는 m대의 로봇을 침입자를 기준으로 $(\pm x, \pm y)$의 4방향으로 로봇을 분배하고, 각 방향의 위치 좌표의 개수와 로봇의 수를 맞춰주었다. m대의 로봇을 각 방향으로 분배하는 방법은 식(2)로부터 얻어낸 탈출이 가능한 장소로 이동할 확률로서 다음과 같이 구할 수 있다.

Step ③ :

for $i = 1$ *to* n

$$\begin{bmatrix} x_{test(n)} \\ y_{test(n)} \end{bmatrix} = \begin{bmatrix} x_{invader} \\ y_{invader} \end{bmatrix} - \begin{bmatrix} x_{site(n)} \\ y_{site(n)} \end{bmatrix}$$

if $x_{test(n)} \geq 0$

if $y_{test(n)} \geq 0$ *then* $P(x+, y+) \mathrel{+}= P(site(n))$

else $P(x+, y-) \mathrel{+}= P(site(n))$

else

if $y_{test(n)} \geq 0$ *then* $P(x-, y+) \mathrel{+}= P(site(n))$

else $P(x-, y-) \mathrel{+}= P(site(n))$

end

위의 과정을 통해 얻은 $P(\pm x, \pm y)$에 로봇의 대수 m을 곱하여 침입자로부터 각 방향에 위치할 로봇의 대수를 결정한다. 이와 같이 결정된 4 방향의 위치와 로봇의 대수가 같지 않으면, 그림 3과 같은 방법을 통해 위치 좌표를 추가하고 줄이면서 로봇의 대수와 같게 해준다. 그림 3(a)는 위치 좌표를 줄이는 방법으로 침입자를 기준으로 두 좌표의 내분점을 계산하여 줄여주며, (b)는 추가하는 방법으로 역시 침입자를 기준으로 같은 길이의 두가지 기준 벡터와의 내분점도 계산하여 추가해준다. 이렇게 구한 $[x_{location(m)}, y_{location(m)}]^T$의 좌표들은 아직 장애물을 고려하지 않은 좌표이다. 그러므로 우리는 이 좌표들이 장애물에 위치하는 좌표인지 체크해주어야 한다. 따라서 마지막으로 ④와 같은 과정을 통하여 로봇의 최종 포메이션 좌표 $[x_{location(m)}, y_{location(m)}]^T$를 구한다.

Step ④ :

for $i = 1$ *to* m

만약 $[x_{location(i)}, y_{location(i)}]^T$가 장애물 좌표이면

while(1)

$$\begin{bmatrix} x_{o(i)} \\ y_{o(i)} \end{bmatrix} = \begin{bmatrix} x_{location(i)} \\ y_{location(i)} \end{bmatrix} - \begin{bmatrix} x_{invader} \\ y_{invader} \end{bmatrix}$$

if $x_{o(i)} > 0$ *then* $x_{location} - 1$

else $x_{o(i)} \leq 0$ *then* $x_{location} + 1$

$x_{location(i)} = x_{o(i)}$

$[x_{location(i)}, y_{location(i)}]^T$가 장애물 좌표가 아니면 *break*

if $y_{o(i)} > 0$ *then* $y_{location} - 1$

else $y_{o(i)} \leq 0$ *then* $y_{location} + 1$

$y_{location(i)} = y_{o(i)}$

$[x_{location(i)}, y_{location(i)}]^T$가 장애물 좌표가 아니면 *break*

①~④의 과정을 통하여 얻은 $[x_{location(m)}, y_{location(m)}]^T$은 로봇의 최종적인 포메이션 좌표이며, 이 좌표를 가지고 다음 경로 계획 알고리즘에서 각 로봇의 경로를 최적화하여 계획해 준다.

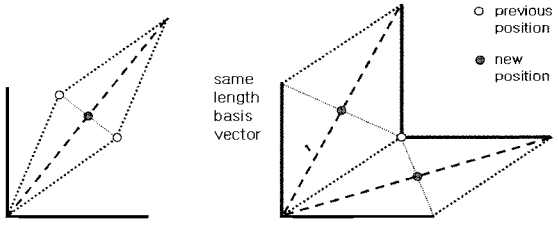

그림 3. (a) 위치 좌표를 줄이는 방법 (b) 위치 좌표를 추가하는 방법

Fig 3. (a) The method for decreasing the formation coordinate (b) The method for increasing the formation coordinate

4. 경로 계획 알고리즘

경로 계획 알고리즘은 앞에서도 언급하였듯이 포메이션 알고리즘으로부터 얻어낸 로봇의 최종 포메이션까지의 경로를 계획하여 주는 알고리즘이다. 우리는 로봇의 최적의 이동 경로를 찾기 위해서 유전 알고리즘(GA)를 사용하였다. 이를 위해서 우리는 주어진 2차원 맵을 각 영역으로 그림 4의 (a)와 같이 각 영역을 직관적으로 구분하여 줌으로써 TSP(Traveling Salesman Problem)문제와 비슷하게 만들어 주어 최적해를 찾아내었다. 이러한 맵을 영역으로 구분하여 주는 것은 맵에 위치하고 있는 장애물에 영향을 받는다. 맵에 위치하고 있는 장애물들로서 생기는 각 코너에 따라서 맵 영역의 수가 결정된다. 그림 4의 (b)는 (a)그림을 어떻게 유전 알고리즘에 어떻게 TSP문제와 비슷하게 적용시킬지 좀 더 알아보기 쉽도록 나타낸 것이다. 다시 말해서 그림4의 (b)와 같이 영역 1에서는 영역 2와 영역 10으로 밖에 갈수가 없다. 각 영역은 해집단의 각 개체들의 적합도 계산에 쓰이기 위해서 다음 식(3)과 같이 적합도를 가진다.

Fitness(영역n) =
(영역 n에 속해있는 탈출 가능한 장소의 적합도)×4 + (영역 n과 바로 연결된 영역에 속해있는 장소의 적합도)×2+ (영역 n에서 다른 영역을 한번 거쳐서 갈 수 있는 영역에 속해 있는 장소의 적합도) (3)

예를 들어, 식(3)을 가지고 그림 4의 영역1의 적합도를 구해보면 다음과 같다.

$Fitness(\text{영역}1)=5\times4+1\times2+(1+1+6)\times1=32$

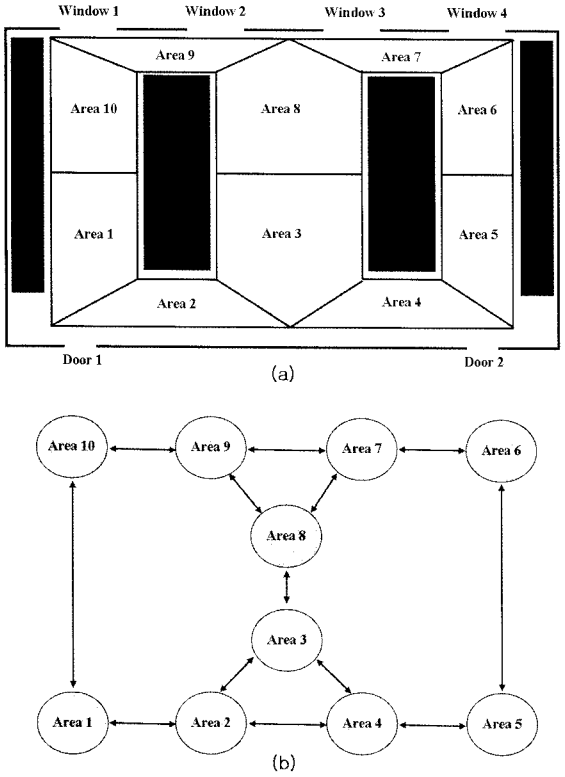

그림 4. (a) 직관적으로 주어진 맵을 영역들로 구분한 예 (b) (a)와 같이 구분된 영역들의 연결 그림

Fig 4. (a) An example for the areas of distributed 2D map (b) The figure of the linked areas as (a)

이렇게 계산된 각 영역의 적합도를 가지고, 경로계획 알고리즘은 다음과 같은 과정으로 수행된다.

Step ① : 현재 로봇의 위치를 받아 현재 어느 영역에 속해 있는지 매핑한다.

Step ② : 랜덤하게 해집단의 개체들을 생성한다. 각 개체들은 각 로봇의 초기 영역위치에서 포메이션 좌표의 영역까지의 경로를 생성하며, 각 영역에서 한번에 갈 수 없는 영역으로는 경로가 생성되지 않는다.

Step ③ : 다음 과정은 각 개체들의 적합도를 부여하기 위한 방법이다. 먼저 어떤 한 영역을 어떤 로봇이 경로로서 선점하면 이 영역의 식(4)와 같이 감소한다.

$$Fitness(\text{영역}n)=\frac{\text{해당 영역을 채택한 로봇의 수}}{\text{총 로봇의 수}} \quad (4)$$

이렇게 영역의 적합도를 감소시키는 이유는 같은 위치에서 몇 대의 로봇이 출발한다면 이 로봇들은 모두 같은 경로를 택하게 될 것이다. 하지만 적합도가 낮은 다른 경로를 아예 무시하여서는 안된다. 만약 로봇이 아니라 사람이라면 사람들은 직관적으로 이러한 사실들을 알고 몇몇은 흩어져서 다른 경로로써 침입자를 포위해 나갈 것이다. 이러한 것을 로봇이 알게 하기란 매우 어려운 일이다. 따라서 우리는 식(4)와 같이 적합도를 감소하게 함으로써 목적지까지 조금 돌아가는 길이라 할지라도 소수의 로봇은 이런 경로를 선택하게 만들어주었다. 이렇게 로봇들의 경로에 해당하는 영역들의 적합도를 가지고 다음 식(5)와 (6)을 통해 각 개체의 적합도가 계산되어진다.

$$Fitness(Robot_n)=(Fitness(1st\ Area)\times1+\ldots+Fitness(ith\ Area)\times\frac{1}{i})\times\frac{1}{i} \quad (5)$$

$$Fitness(individual_i)=\sum_{n=1}^{m}Fitness(Robot_n) \quad (6)$$

식(6)과 같이 각 개체의 적합도는 각 로봇들의 적합도의 합으로 이루어지며, 로봇들의 적합도는 식(5)와 같이 경로로서 선택된 영역의 적합도의 합에 경로의 수로서 나누어 결정된다. 따라서 경로의 수가 많아질수록 적합도는 감소하게 되어 최적화된 경로를 찾을 수 있게 해준다.

Step ④ : 각 개체들의 변이를 통해 다음 세대의 새로운 해집단을 생성한다. 여기서 교차는 해당 영역들의 연결에 의해서 해집단의 개체가 생성되므로 배제한다.

Step ⑤ : 종료 조건을 만족할 때까지 ①~④과정을 반복한다.

위의 과정을 통하여 얻어낸 각 로봇의 이동 경로는 아직 개략적으로 어떤 영역을 어떤 순서로 통해서 최종 포메이션 좌표까지 이동할 것인가를 결정해준 것 일뿐 로봇의 정확한 이동 경로를 결정하여 주지는 못한다. 따라서 우리는 위의 과정을 통하여 얻어낸 로봇의 이동 경로를 가지고 로봇의 정확한 이동 경로를 다음과 같은 과정을 통해 결정해주었다.

Step ① : 선택된(적합도가 가장 높은) 개체의 로봇들의 이동 경로에서 같은 순서에 같은 영역에 도착하는 로봇들의 수를 체크한다.

Step ② : 각 영역마다 ①의 과정을 통해 체크한 각 로봇의 수를 가지고 그림 5의 (a)와 같은 방법으로 로봇이 지나갈 각 영역의 포인트를 생성한다. 이 포인트는 로봇이 최단경로로 이동하게 하기 위하여 코너의 안쪽부터 포인트의 좌표가 미리 정해져 있으며, 로봇의 크기보다 큰 간격으로 차례로 생성된다. 그림 5의 (a)는 같은 순서에 영역 2에서 영역 1로 이동하는 로봇이 2대 있을 때 생성되는 예를 나타낸 그림이다.

Step ③ : ②의 과정을 통해 생성된 포인트와 로봇의 위치와의 유클리디안 거리(Euclidean Distance)를 구하여, 해당 영역의 로봇들의 경로의 거리가 짧은 거리의 경로를 그림5의 (b)와 같이 로봇에게 분배한다. 이 때 로봇에 가까운 쪽에 생성된 포인트와 먼 쪽에 생성된 포인트중에서 먼 쪽에 생성된 포인트까지의 직선상에 장애물이 위치하지 않을 때에는 먼 쪽의 포인트가 선택되어 앞의 과정을 거친다. 이 과정은 마지막 로봇의 최종 포메이션 위치 좌표까지 순차적으로 유클리디안 거리를 구하여 로봇에게 분배된다.

이렇게 결정된 로봇의 이동 경로로서 시스템은 각 로봇에게 첫 포인트까지의 초기 이동 방향을 전송하여 주며, 지속적으로 로봇의 현재 위치를 전송받아 해당 포인트까지 도착을 완료하면 다음 포인트까지의 이동 방향을 전송하여 최종 포메이션 좌표까지의 이동을 명령한다.

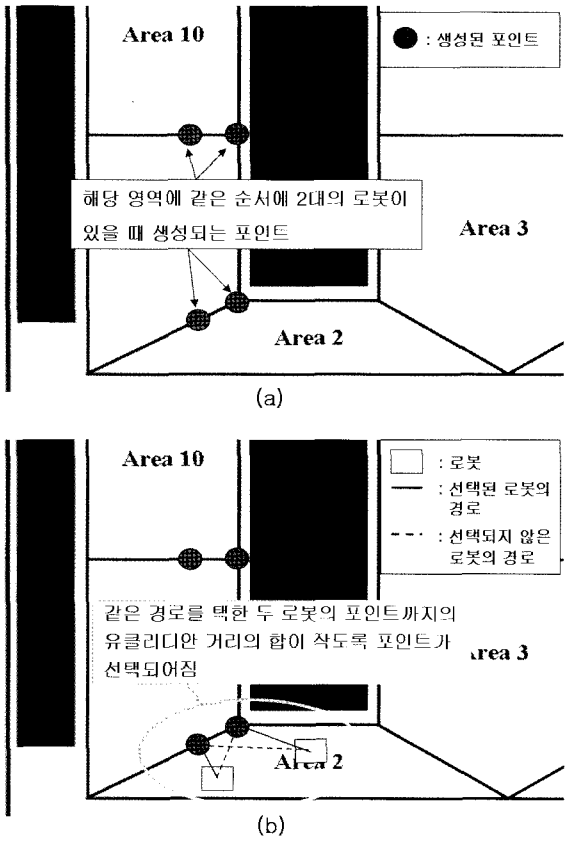

그림 5. (a) 같은 영역에 동시에 2대의 로봇이 있을 때 생성되는 경유 좌표의 예 (b) 경유 좌표를 로봇에게 할당하는 방법

Fig 5. (a) When two robot simultaneously is existed a same area, an example for the intermediate point which is created (b) The method for allotting the intermediate point to each robot

5. 시뮬레이션

본 논문에서는 시뮬레이션을 통해 제안된 알고리즘의 성능을 평가하였다. 시뮬레이션에서 로봇의 수는 10대로 하였으며, 경로 계획 알고리즘에 사용된 유전 알고리즘의 해집단의 개체수는 50, 변이 확률은 0.8, 종료 조건은 200세대로 실험을 진행하였으며, 그림 4(a)와 같은 맵에서 로봇과 침입자의 위치를 바꾸어 가며 시뮬레이션을 진행하였다. 로봇의 각 경로에 위치한 중간점이 생성된 포인트이며, 침입자를 포위하고 있는 점이 각 로봇의 최 종 위치이다. 그림 6은 로봇의 초기 위치는 양쪽 문이며, 침입자는 중앙 부근에서 발견되었을 때의 실험으로 그림상에서 왼쪽에 위치한 1대의 로봇과 오른쪽에 위치한 2대의 로봇이 먼 경로를 통하여 포위 위치까지 이동한 것을 볼 수 있다. 그림 7은 발견된 침입자의 위치는 실험 1과 같지만 로봇들의 초기 위치가 각 영역에 산재되어 있을 때의 실험 결과이다. 왼쪽위 영역에 3대의 로봇이 존재하였기 때문에 적합도는 낮지만 경로가 짧아 첫 번째 실험과는 다르게 2대의 로봇이 위쪽 경로를 택하였다. 그림 8은 로봇의 초기 위치는 양쪽 문이며, 왼쪽 영역에서 침입자가 발견되었을 때의 실험 결과이며, 그림 9는 로봇의 초기 위치는 역시 양쪽 문이며, 오른쪽 영역에서 침입자가 발견되었을 때의 실험 결과이다. 역시 그림 8과 9의 두 결과 모두 로봇의 최종 포메이션과 분산 이동 경로가 만족할만한 결과임을 알 수 있다. 우리는 이와 같은 실험 결과로서 본 논문에서 제안한 알고리즘이 기대되는 성능을 만족한다는 것을 알 수 있었다.

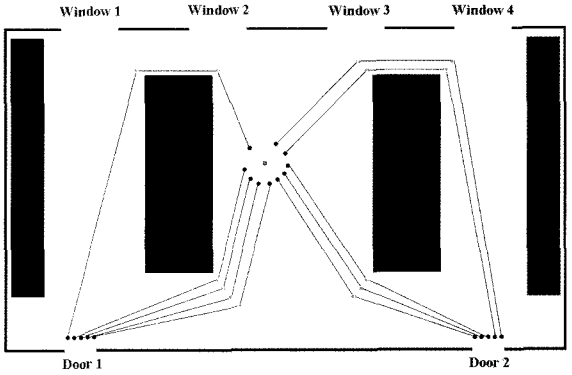

그림 6. 군집 로봇의 침입자 포위 실험 1

Fig 6. A 1st experiment on enclosing an invader for swarm robots

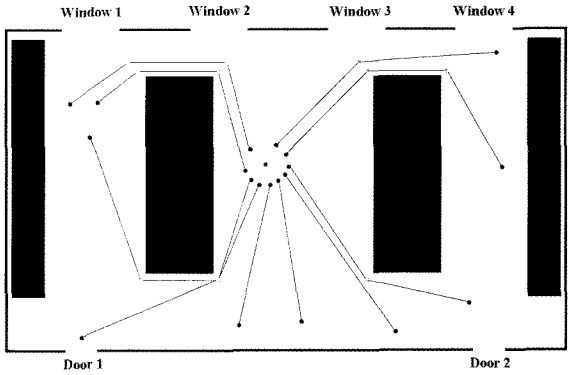

그림 7. 군집 로봇의 침입자 포위 실험 2

Fig 7. A 2nd experiment on enclosing an invader for swarm robots

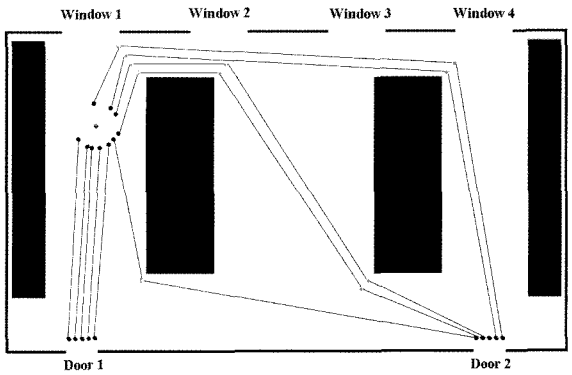

그림 8. 군집 로봇의 침입자 포위 실험 3

Fig 8. A 3rd experiment on enclosing an invader for swarm robots

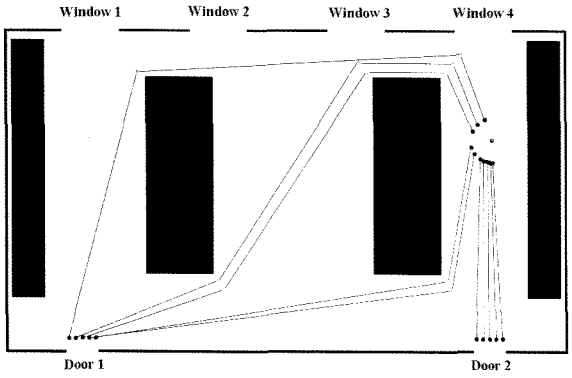

그림 9. 군집 로봇의 침입자 포위 실험 4
Fig 9. A 4th experiment on enclosing an invader for swarm robots

6. 결 론

본 논문에서는 침입자 포위를 위한 군집 로봇의 분산 이동 알고리즘에 대해 제안하였고, 시뮬레이션을 통해 알고리즘의 성능을 평가하였다. 우리는 침입자 발견 시 로봇들이 침입자를 효과적으로 포위하기 위해서 주어진 2차원 맵의 탈출 가능한 장소에 확률을 테이블화하여 가짐으로써 침입자에서 탈출 가능한 장소로의 방향 벡터를 구하고, 이를 기준으로 로봇의 최종 포메이션 좌표를 결정하였으며, 결정된 좌표까지의 이동시 효과적으로 침입자를 탐색하며 이동하기 위해서 경로 계획 알고리즘에서는 로봇의 전략적인 분산 이동 경로를 계획하여주었다. 시뮬레이션 결과를 살펴봄으로써 본 알고리즘이 기대한 결과와 비슷한 성능을 보여주는 것을 알 수 있었다. 하지만 이것이 최적의 알고리즘이라고는 할 수 없으며, 현실에서는 여러 가지 변수들이 존재하기 때문에 성능을 보장할 수 없다. 또한 주어진 2차원 맵이 방대해질수록 또 로봇의 대수가 많아질수록 계산량의 증가에 따라 시스템의 속도가 현저하게 낮아질 수 있다. 따라서 이러한 환경 변수에 강인할 수 있도록 알고리즘을 더욱 개선하여야 할 것이다. 이를 위해서 앞으로는 컴퓨터 시뮬레이션 상이 아닌 실제 로봇을 가지고 실험을 하여야 할 것이다. 또한 우리가 임의로 정해준 여러 가지 변수들, 예를 들어 탈출이 가능한 장소의 적합도, 영역의 적합도와 개체들의 적합도 계산 등을 학습을 통하여 더 나은 성능을 보여줄 수 있도록 하여야 하며, 방대한 계산량을 줄일 수 있도록 좀 더 간단한 형태를 고려해보아야 할 것이다.

참 고 문 헌

[1] D. J. Pack and B. E. Mullins, "Toward Finding an Universal Search Algorithm for Swarm Robots," *in Proc. 2003 IEEE/RSJ Int. Conference on Intelligent Robots and Systems,* pp. 1945-1950, 2003.

[2] G. Antonelli and S. Chiaverini, "Kinematic control of a platoon of autonomous vehicles," *in Proc. 2003 IEEE Int. Conf. Robotics and Automation,* Taipei, pp. 1464-1469, 2003.

[3] G. Antonelli and S. Chiaverini, "Kinematic control of platoons of autonomous vehicles," *IEEE Trans. Robot,* vol. 22, pp. 1285-1292, Dec. 2006.

[4] H. Yamaguchi, "A Cooperative Hunting Behavior by Mobile Robot Troops," *in Proc. 1998 IEEE,* Leuven, Belgium, pp. 3204-3209, 1998.

[5] H. Yamaguchi, "A Cooperative Hunting Behavior by Multiple Nonholonomic Mobile Robots," *in Proc. 1998 IEEE,* pp. 3347-3352, 1998.

[6] P. Pirjanian and M. Mataric, "Multi-robot target acquisition using multiple objective behavior coordination," *in Proc. 2000 IEEE Int. Conf. Robotics and Automation,* vol. 3, pp. 2696-2702, 2000.

[7] T. Kamano, T. Yasuno, T. Suzuki, Y. Hasegawa, H. Harada and Y. Kataoka, "Design and implementation of fuzzy cooperative catching controller for multiple mobile robots," *in Proc. 26th Ann. Conf. of the IEEE Industrial Electronics Society,* pp. 1749-1754, 2000.

저 자 소 개

이희재(Hee-Jae Lee)
2003년 : 중앙대학교
전자전기공학부 공학사
2007년 ~ 현재 : 중앙대학교 대학원
전자전기공학부 석사과정

관심분야 : 최적화문제, 진화연산, 소프트 컴퓨팅, 패턴 인식 등

심귀보(Kwee-Bo Sim)
1990년 : The University of Tokyo
전자공학과 공학박사
1991년 ~ 현재 : 중앙대학교
전자전기공학부 교수

[제19권 제1호 (2009년 2월호) 참조]

E-mail : kbsim@cau.ac.kr
Homepage URL : http://alife.cau.ac.kr